문자열

문자열에 대해 전반적으로 알아보는 파트입니다.

- 더글라스의 주관이 많이 들어간듯 한 부분과 더글라스가 직접 구현한 fullfill 함수 부분은 정리에서 제외했습니다.

자바스크립트의 문자열.

- 자바스크립트에서 문자열은 16비트(2바이트) 배열
  - 16비트로 문자열을 다뤄서 나중에 말할 유니코드 문제가 발생..

- 배열 하나하나의 요소를 캐릭터코드(`char code`) 라고합니다.
- `char code` 의 배열을 `String.fromCharCode` 로 문자열로 만들 수 있고, 문자열에서 charCodeAt 함수로 `char code`에 접근할 수도 있습니다.
- 문자열에 [] 연산자로 문자 개별 값을 읽어올 수 있습니다.
  - 이 경우 char code 를 가져오지는 않고, 길이가 1인 새로운 문자열을 반환합니다.
- 자바스크립트에서의 문자열 동등성 검사는, `값이 같을 경우` `===` 연산자에 의해 동일하다고 평가됩니다.
  - 이는 아주 강력한 기능입니다(앞서 `charcode`의 배열이라 했지만, 배열과 다르게 값으로 동등성을 평가합니다.)
  - 많은 언어에서 문자열은 각자의 참조를 가지는 경우가 많은데, 자바스크립트는 그렇지 않습니다.

유니코드

- 유니코드를 다루면서 아주 고통을 맛본적이 많습니다... 책에서 잘 설명 되어있네요. 한번 같이 알아보도록 합시다.

유니코드의 구성

- 유니코드는 원래 모든 언어를 16비트로 표현하려는 시도였습니다. 하지만 불행하게도... 전 세계의 언어를 21비트로 표현하는 스펙으로 바뀌어 버렸습니다.
- 그리고 정말로 불행하게도, 앞서 말했듯 자바스크립트 문자열은 16비트 배열입니다.
- 그렇기때문에 자바스크립트 문자열 하나로 표현할 수 없는 유니코드가 존재하게 되었습니다.

- 유니코드는 자바스크립트 문자를 받아 `코드유닛`, `코드포인트` 로 나눕니다.
- 코드 포인트는 특정 문자에 해당하는 숫자이며 총 `1114112` 개 입니다.
- 2바이트의 크기는 `65536` 이기때문에, 하나의 코드 포인트 그룹을 `평면` 이라고 칭하고, 총 17개의 평면이 나오게 됩니다(`65536 * 17 = 1114112`).

자바스크립트에서 유니코드를 다루는 방법

- 자바스크립트는 16비트로 표현할 수 없는 유니코드에 대해 대리쌍(surrogate pairs)라는 방식을 사용했습니다.
- 대리쌍은 2개의 특별한 코드 유닛으로 구성됩니다.
- 상위 대리쌍은 `0xD800`, 하위 대리쌍은 `0xDC00` 의 오프셋을 가집니다.

현실에서 마주할 수 있는 문제

- 저 이상한 유니코드를 알아서 무엇하리 싶지만, 불행하게도 인스타 감성의 보급으로 사람들은 이모지를 아주아주 많이 사용하기 시작했습니다.
- 😭 (눈물...)
- 이 눈물의 length는 얼마일까요?

```
console.log("😭".length) // 2
```

- 왜 2인지는 이제 예측할 수 있을것입니다...
- 눈물의 유니코드 값이 `U+1F62D` 이기 때문입니다 (16비트를 넘어선것이죠..)

- 해당 유니코드는 2개의 대리쌍으로 표현됩니다.

```
"😭".charCodeAt(0).toString(16) // 'd83d'
"😭".charCodeAt(1).toString(16) // 'de2d'
`\ud83d\ude2d` // "😭"
```

- 원리를 간단히 정리해보자면, U+1F62D -> 128557 (10진수)
- 128557 - 65536 = 63021
- 이 값을 10비트씩 자르면, `3D`, `22D`
- 오프셋을 각각 더하면
  - 3D + D800 = D83D
  - 22D + DC00 = DE2D

문제를 마주하지 않으려면?

- 다행히도 es6 이후에는 유니코드를 다루기가 그나마 수월해졌습니다.

```
[...'😭'].length //1 감사합니다..
const emojiString = "😭😭😭"
for(let elem of emojiString){
    console.log(elem)
} // 😭😭😭

for (let i = 0 ; i<emojiString.length; i++){
    console.log( emojiString[i])
} // ????? 재앙...
```

- es6를 활용할 수 있는 환경에선 괜찮지만, 그렇지 않다면 기획자와 충분한 대화!(오늘도... 개발자가... 안된다그랬다.)

템플릿 리터럴

- 읽어보니 더글러스는 별로 좋아하지 않는 기능같은데 저는 아주 좋아하는 기능입니다.
- const exampleStr =`hello world` 와 같이 사용합니다.
- 문자열 보간을 사용할 수 있습니다.
  - const hoho = "호호호 자따구"
  - const exampleStr =`hello world ${hoho}`

- 태그드 템플릿이라는 기능을 제공합니다.

```
const pw = "12345"
const sk="abc987"
let t1 = censorTemplete`this is my pw ${pw} and secret ket is ${sk}`
const censor = (str) => "*".repeat(str.length)
const censorTemplete = (str,...arg) => {
  // 문자열 보간에 의해 잘린 토큰들이 str배열로 들어옴
  // str = ["this is my pw ", " and secret ket is ", ""]
  // arg = [12345,abc987]

  return str.reduce(
    (acc,e,i) => `${acc}${e}${censor(arg[i]||'')}`
  ,'')
} // 이렇게 하면 자동으로 비밀번호가 *** 처리되는 함수를 만들 수 있음.
```

자바스크립트의 빈값

- 자바스크립트에서는 `null` 과 `undefined` 라는 두개의 빈 값이 있습니다.
- 두 값에는 차이가 있지만, 둘 중 하나만 사용하는 편이 낫습니다(특히 ts에서 귀찮죠..).
- 저도 더글라스도 `undefined` 를 사용하는것을 선호합니다
  - `typeof null` 은 `object` 입니다(진짜 이상한 자바스크립트..)
  - 다만 빈 객체를 만들 때 Object.Create(null) 을 사용하는 경우는 제외합니다.
  - undefined를 넣거나 값 없이 부를 수 없는 함수입니다.

- 자바스크립트의 null 과 undefined 는 외국에서도 많은 토론이 이루어지고 있는듯 합니다.
- https://github.com/sindresorhus/meta/discussions/7
- json 은 undefined 를 지원하지 않습니다
- db에서 null 인 값을 어떻게 표현할것인지?
- react의 ref도 기본값이 null,
  `type Ref<T> = { current: null | T }. null`
- 어떤 라이브러리는 빈값을 null 로 쓰고 어떤 라이브러리는 undefined로 쓰고... 우리가 해결할 수 있는 문제는 아니지만 통일시켜 줬으면 좋겠네요.